(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
14 December 2006 (14.12.2006)

PCT

(10) International Publication Number
WO 2006/132639 A1

(51) International Patent Classification:
G06F 17/50 (2006.01)

(21) International Application Number:
PCT/US2005/020242

(22) International Filing Date: 7 June 2005 (07.06.2005)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant *(for all designated States except US)*: **THE REGENTS OF THE UNIVERSITY OF CALIFORNIA** [US/US]; 1111 Franklin Street, 5th Floor, Oakland, CA 94607-5200 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **CHENG, Chung-kuan** [US/US]; 4407 Mensha Place, San Diego, CA 92130 (US). **ZHU, Zhengyong** [CN/US]; 9152 F Regents Road, La Jolla, CA 92037 (US). **SHI, Rui** [CN/US]; 9248 Regents Road, Apt#A, La Jolla, CA 92037 (US).

(74) Agent: **AI, Bing**; Fish & Richardson P.C., 12390 El Camino Real, San Diego, CA 92130 (US).

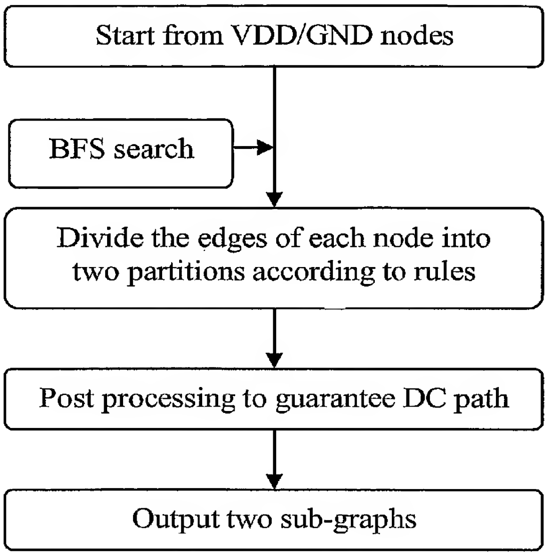
(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: CIRCUIT SPLITTING IN ANALYSIS OF CIRCUITS AT TRANSISTOR LEVEL


Start from VDD/GND nodes
BFS search
Divide the edges of each node into two partitions according to rules
Post processing to guarantee DC path
Output two sub-graphs

(57) Abstract: Analyzing the integrated circuit using node model and dividing the node edges to form two sub graphs that represent two sub circuits of the integrated circuit with operating splitting method figure 3 for splitting and analyzing the two sub circuits for improved computation efficiency and improved processing speed.

WO 2006/132639 A1

CIRCUIT SPLITTING IN ANALYSIS OF CIRCUITS AT TRANSISTOR LEVEL

Background

[0001] This application relates to analysis of integrated circuits.

[0002] Circuits may be viewed as networks of nodes and circuit elements connected between nodes. As such, circuits may be analyzed based on a nodal analysis where a nodal equation may be written for each node based on the conservation of charge at the node, i.e., the total current entering the node is equal to the total current leaving the node (the Kirchoff's second rule). For a circuit with N nodes, N equations for the N nodes can be expressed in terms of the properties of circuit elements such as resistance, capacitance, and inductance, and in terms of the node voltages and currents. These N equations can be written into a matrix equation and are solved using various matrix approaches such as LU decompositions.

[0003] Integrated circuits with transistors can be simulated using direct methods such as LU decompositions. One example, the Berkeley SPICE2 simulator and its variations use LU decompositions to solve for circuit equations for circuits with transistors. See, Nagal, "Spice2: A computer program to simulate semiconductor circuits," Tech. Rep. ERL M520, Electronics Research Laboratory Report, UC Berkeley (1975). The direct simulation methods may become less effective and can reach their computational limits when the number of transistors and other elements in circuits approaches the capacity limit, e.g., around 50,000 transistors for some direct simulation methods. This is in part because the super linear complexity O($n^{1.5}$) increases with the number of circuit nodes, n, and the amount of the extracted interconnect data for a large n can exceed the capacity of the software based on a direct simulation method.

Summary

[0004] This application describes, among others, a general operator splitting technique for analyzing large-scale integrated circuits to provide ensured convergence and improved simulation speed with high accuracy. The general operator splitting technique can be used to reduce the amount of non-zero fill-ins during the LU factorization. Various implementations of the splitting are described.

[0005] In one implementation, a method for analyzing an integrated circuit includes splitting the integrated circuit into two sub circuits which are in or close to tree or forest structures and have a DC path to GND or VDD for each node; and applying a direct matrix inversion method to a circuit matrix equation for each of the two sub circuits to find solutions to the sub circuits. Accordingly, an article is also described to include at least one machine-readable medium that stores machine-executable instructions for the above method. The instructions cause a machine to split the integrated circuit into two sub circuits which are in or close to tree or forest strcutrues and have a DC path to GND or VDD for each node; and apply a direct matrix inversion method to a circuit matrix equation for each of the two sub circuits to find solutions to the sub circuits.

[0006] In another implementation, a method for analyzing an integrated circuit is described to represent the integrated circuit with a graph. The graph is divided into two subgraphs that respectively represent two sub circuits of the integrated circuit. In the division process, a resistor is divided evenly between the two subgraphs while a transistor, a transistor device, a capacitor, or an inductor is duplicated in the two subgraphs. Each node of each subgraph has a DC path to GND or VDD. The solutions to the two subgraphs are found for analyzing the integrated circuit. Accordingly, an article is described to include at least one machine-readable medium that stores machine-executable instructions for the above method.

[0007] In yet another implementation, an integrated circuit under analysis is represented by a graph. From one or more nodes connected to GND or VDD in the graph, a search for nodes is conducted in the graph. The edges of nodes from the search are divided to form two subgraphs that respectively represent two sub circuits of the integrated circuit. The two subgraphs are modified to ensure each node of each subgraph to have a DC path to GND or VDD. The two subgraphs are analyzed to find solution to the integrated circuit. Accordingly, an article is described to include at least one machine-readable medium that stores machine-executable instructions for the above method.

[0008] These and other implementations and their applications are described in greater detail in the attached drawings, the following detailed description, and the claims.

Brief Description of Drawings

[0009] FIGS. 1A and 1B show one example of a simple transistor circuit and its corresponding graph, respectively.

[0010] FIGS. 2A, 2B and 2C illustrate one example of the splitting of the circuit in FIG. 1A according to one implementation.

[0011] FIG. 3 illustrates one implementation of the splitting process.

[0012] FIGS. 4A, 4B, 4C, 4D, 4E, 4F and 4G show a step-by-step illustration of the application of the splitting in FIG. 3 for a simple 6x6 mesh transistor circuit.

[0013] FIGS. 5A, 5B, 5C and 5D show simulation results using the present splitting technique.

Detailed Description

[0014] The circuit splitting described here partitions the circuit equation for a complex integrated circuit into two less complicated subcircuits to allow for efficient computations of the circuit equations for the two subcircuits and to provide ensured convergence. The subcircuits have fewer elements than the original circuit. The splitting concept was introduced as a technique for solving partial differential equations. See, e.g., W. F. Ames, "Numerical Methods for Partial Differential Equations," 2nd edition, New York Academic Press (1977). This operator splitting can be illustrated with the following initial value problem (IVP) of a simple ordinary differential equation (ODE),

$$\frac{\delta u}{\delta t} = Lu \qquad (1)$$

where L is a linear or nonlinear operator and can be written as a linear sum of m suboperators L_1, L_2,…, and L_m of u,

$$Lu = L_1 u + L_2 u + \cdots + L_m u \qquad (2)$$

Suppose u_1, u_2, ..., u_m are updating operators on u with respect to L_1, L_2, • • •, L_m from time step n to time step $n + 1$, the operator splitting approach has the form of:

$$\begin{aligned} u^{n+(1/m)} &= U_1(u^n, h/m) \\ u^{n+(2/m)} &= U_2(u^{n+(1/m)}, h/m) \\ &\cdots \\ u^{n+1} &= U_m(u^{n+(m-1)/m}, h/m \end{aligned} \qquad (3)$$

where each partial operation acts with all the terms of the original operator.

[0015] The circuit splitting described here uses the graph theory as the basis to generate a graph representation of an integrated circuit under analysis and then applies a set of partition rules to split the graph for the original circuit into two subgraphs respectively representing two smaller circuits. Circuits nodes are represented by vertices of graphs and circuit connections between different nodes are represented by edges in graphs. This use of the graph based modeling for circuits can be used to provide a generalized splitting algorithm and the generalization removes constraints of the geometry or locality of circuits under analysis. Notably, this approach is unconditionally stable and hence essentially eliminates the convergence issues in many direct methods for simulating large and complex circuits such as the LU decompositions.

[0016] The following sections use a general circuit system to describe the circuit operator splitting method. It is assumed that the circuit includes resistors, capacitors, and inductors with mutual couplings. For linear circuits, the nodal analysis using Backward Euler Integration can be expressed as below:

$$\begin{bmatrix} \frac{C}{h}+G & -A^T \\ A & \frac{L}{h}+R \end{bmatrix} \begin{bmatrix} V(t+h) \\ I(t+h) \end{bmatrix} = \begin{bmatrix} \frac{C}{h} & O \\ O & \frac{L}{h} \end{bmatrix} \begin{bmatrix} V(t) \\ I(t) \end{bmatrix} + U(t+h) \qquad (4)$$

where C, L, R, G are the matrices of capacitances, inductances, resistances, and conductances, respectively; the matrix A is an incidence matrix linking between the topology of capacitance nodes and inductance branches; the vectors V, I, and U describe the voltages of capacitance nodes, currents of inductance branches, and system inputs, respectively. The scalar h is the time step from time *t* to time *(t+h)*. Note that the four matrices, C, L, R, and G, are symmetric by construction and are positive semidefinite because the circuit elements such as capacitances, inductances, resistances, and conductances are non-active. In addition, matrices C and L are assumed to be positive definite for a nondegenerated case.

[0017] The circuit represented by Equation (4) can be partitioned into any two arbitrary subcircuits. The corresponding two partitions of matrices A, R, and G can be written as:

$$A = -A_1 + A_2,$$
$$R = R_1 + R_2 \text{ and}$$
$$G = G_1 + G_2.$$

By construction, the matrices Ri and Gi for $i \in \{1,2\}$ are symmetric and positive semidefinite. Following the circuit partition, the integration in Equation (4) is divided into two half steps and alternates the forward and backward integrations between the partitions as follows:

$$\begin{cases} \begin{bmatrix} \frac{2C}{h}+G_1 & -A_1^T \\ A_1 & \frac{2L}{h}+R_1 \end{bmatrix} \begin{bmatrix} V\left(t+\frac{h}{2}\right) \\ I\left(t+\frac{h}{2}\right) \end{bmatrix} = \\ \qquad \begin{bmatrix} \frac{2C}{h}-G_2 & A_2^T \\ -A_2 & \frac{2L}{h}-R_2 \end{bmatrix} \begin{bmatrix} V(t) \\ I(t) \end{bmatrix} + U\left(t+\frac{h}{2}\right) \\ \begin{bmatrix} \frac{2C}{h}+G_2 & -A_2^T \\ A_2 & \frac{2L}{h}+R_2 \end{bmatrix} \begin{bmatrix} V(t+h) \\ I(t+h) \end{bmatrix} = \\ \qquad \begin{bmatrix} \frac{2C}{h}-G_1 & A_1^T \\ -A_1 & \frac{2L}{h}-R_1 \end{bmatrix} \begin{bmatrix} V\left(t+\frac{h}{2}\right) \\ I\left(t+\frac{h}{2}\right) \end{bmatrix} + U(t+h) \end{cases} \qquad (5)$$

In the first half step, the forward integration is used for the subcircuit with matrices A_2, G_2 and R_2. Then, in the second half step, the forward integration is used for the subcircuit with matrices A_1, G_1 and R_1. In both half steps, the other partition is integrated by the backward implicit integration.

[0018] If the two left-hand-side matrices in the two equations in the formulation (5) correspond to trees or forest structures, a direct matrix inversion can be used to efficiently solve those two equations because there is no nonzero fill-ins and the computational cost is linearly proportional to the number of elements. Define notations P1, P2 and S as follows:

$$P_1 = \begin{bmatrix} G_1 & -A_1^T \\ A_1 & R_1 \end{bmatrix},$$

$$P_2 = \begin{bmatrix} G_2 & -A_2^T \\ A_2 & R_2 \end{bmatrix}, \textit{ and },$$

$$S = \begin{bmatrix} \frac{2C}{h} & 0 \\ 0 & \frac{2L}{h} \end{bmatrix}.$$

The two equations for the two half steps of the operator splitting formulation (5) can be simplified as:

$$\begin{cases} (P_1+S)X\left(t+\frac{h}{2}\right) = -(P_2-S)X(t)+U\left(t+\frac{h}{2}\right) \\ (P_2+S)X(t+h) = -(P_1-S)X\left(t+\frac{h}{2}\right)+U(t+h) \end{cases} \quad (6)$$

where $X = \begin{bmatrix} V \\ I \end{bmatrix}$.

[0019] The above operator splitting formulation allows for arbitrary splitting of a circuit. The present technique applies a set of splitting rules to ensure more efficient processing of the two partitions and the original circuit. Such splitting rules are in part based on the recognition that the performance of direct methods such as LU decomposition can still beat iterative methods for small circuits with up to tens of thousands of nodes and direct methods become prohibitive for large circuits due to the order of $O(n^{1.1})$ to $O(n^{1.5})$ nonzero fill-ins, where n is the number of nodes in circuits. In addition, it is also known that the LU decomposition method does not create non-zero fill-ins for circuits in tree/forest structure if nodes elimination always selects from leaves of the graph. This elimination order can be captured by ordering algorithms based on minimum degrees (i.e., the number of neighboring nodes for a particular node).

[0020] Following this observation, the present operator splitting algorithm splits a circuit into two partitions with structures close to tree or forest such that the number of non-zero fill-ins is minimized. The partition is carried out in two steps: graph representation of the circuit and application of the splitting rules to form two sub graphs.

[0021] In the implementations described here, a circuit structure can be represented by an undirected graph. The edges in the undirected graph are divided into two sub-graphs using the graph theory. An undirected graph G = (V, E) is used to represent the circuit structure.

[0022] In order to ensure the DC convergence of the two partitions, every node in both partitions has a DC path to GND or VDD. This is one of the rules of the present splitting algorithm. As an example, because each capacitor is a circuit break point in a DC path, there cannot a node between two capacitors. In addition, only resistive connections are split and resistors are divided into two partitions using graph theory algorithms. Capacitors and inductors are not divided so that each partition has a full version of each capacitor or inductor. When solving each partition, the rest of circuit are modeled as equivalent current sources, following the operator splitting formulation (5) or (6). In many digital circuits, transistors are often grouped as various gates. Taking into consideration the nonlinear property of transistor devices and gates, a single transistor or gate is not split into different partitions; instead each partition has a full-version of all transistor devices. Accordingly, transistor devices are treated like capacitors and inductors and thus are duplicated in each partition and are solved at every half time point. Hence, the present splitting algorithm divides each resistor branch into two partitions and each partition has a full-version of transistor devices, capacitors and inductors. Circuit nodes can be classified as super nodes and branch nodes in the graph for the circuit. A super node denotes the end point of resistors in large linear networks or a single gate. Branch node represents the end point of resistors on signal wires connecting gates in the circuit. The edge denotes the resistor branch in the circuit since only resistors are divided into partitions. End points of resisters on signal wires and large linear networks (e.g., a power network) are different. Only end points on signal wires are considered as branch nodes. With the transistor duplication strategy, each gate is represented by a super node in the graph and the detail inside each gate is invisible to the splitting algorithm. The resistors connection points in the circuit are represented by a branch node in the graph. Each edge e_{ij} between nodes v_i and v_j in the graph represents a corresponding connection in the circuit.

[0023] FIG. 1A shows a simple transistor circuit with three inputs and two outputs. FIG. 1B shows the corresponding undirected graph representation of the circuit in FIG. 1A. The five super nodes in solid dots represent five gates in the circuit respectively and four branch nodes in hollow dots represent the resistive connections in the circuit. The outputs are omitted. FIG. 2A shows one possible splitting configuration of the graph in FIG. 1B where dashed lines and solid lines denote two different partitions. FIGS. 2B shows the sub graph of the solid lines in FIG. 2A and FIG. 2C shows the sub graph of the dashed lines in FIG. 2A.

[0024] The splitting algorithm is applied to divide the graph into two sub-graphs. Each sub-graph represents a sub-circuit and corresponds to a sub-matrix in circuit simulation process. The number of non-zero fill-ins of the matrix LU decomposition is to be as small as possible during the partition so that the sum of the number of non-zero fill-ins of two sub-matrixes much less than the number of non-zero fill-ins of the original full matrix. To achieve this, the rules for splitting the graph are designed to decrease the degree (the number of neighbors) of every node in both sub-graphs compared with the degree in the original undirected graph. In the matrix LU decomposition, non-zero fill-ins are introduced among neighbors of a node when the node is eliminated and as such the non-zero fill-ins increase dramatically as the elimination progresses and eventually render the LU decomposition impractical for solving circuits with a large number of nodes. The present splitting algorithm minimizes the degree of every node in sub-graphs and thus reduces the number of non-zero fill-ins.

[0025] In addition to the above rules on the DC paths, resistors, inductors, capacitors, and transistors, the following rules are applied for the splitting process.

[0026] (1) Branch rule: the edges in one branch belong to the same partition. For example, signal wires connecting gates are assigned to one partition. In the undirected graph, one branch includes the edges connected by branch nodes. If one branch is broken into two partitions, the broken branch node could cause undesired iterations during simulation.

[0027] (2) Degree rule: the edges of a node whose degree is two belong to the same partition. The line structure wouldn't cause many non-zero fill-ins and it will be propitious to provide the DC path in the sub-graphs.

[0028] (3) Loop rule: the loop is to be avoided in each sub-graph if possible. Loops in the sub-graphs can potentially introduce non-zero fill-ins and thus should be avoided or minimized if possible.

[0029] (4) Balance rule: the edges for each node in the graph are to be divided into two sub-graphs.

[0030] Application of the above splitting rules may render the partition of a circuit not optimized due to its structure limitation or the restriction of DC paths (no floating nodes is allowed at DC stage). However, the number of overall non-zero fill-ins can still be greatly reduced for most circuits in comparison with the number of non-zero fill-ins due to the direct methods. In addition, the present splitting ensures the stability of the computation based on the splitting.

[0031] FIG. 3 shows one implementation of the splitting algorithm. First, a circuit under analysis is represented by a corresponding undirected graph of super nodes and branch nodes where the VDD and GND nodes are identified. Next, a breadth first search (BFS) is performed beginning at the identified VDD and GND nodes to discover all nodes in the graph. Splitting rules are applied to divide edges of each node of the graph into two partitions. A DC path post processing is then performed to modify the partitions in order to ensure a DC path for each node in the two sub graphs. The final result is two sub-graphs, which correspond to two sub-circuits.

[0032] In the BFS partition stage, the search may begin from VDD/GND nodes simultaneously to go through all the nodes in the graph using BFS and to divide the edges of every node into two partitions according to the partition rules. In order to facilitate the post processing, each of the nodes and edges is associated with a label to record the status of partition and DC path to VDD/GND. Based on the labeling information, the partition rules are applied to benefit the DC path available for all the nodes in sub-graphs.

[0033] In the post processing stage, the partition for the edges without a DC path to VDD/GND is adjusted or modified. According to the labels, the partition status for all the edges and nodes is identified to indicate which partition each edge or node belongs to and whether each edge or node has a DC path to VDD/GND in that partition. From a global view of the BFS partition result, some parts of the graph would have DC paths while other parts of the graph may not have a DC path. Therefore, there are boundary nodes between a part with DC paths and another adjacent part without a DC path. The post processing is to extend or "grow" an existing DC path from a boundary node for the edges and nodes without a DC path upon the partition. For example, if an edge does not have a DC path in the partition assigned to it while one of its connecting nodes has a DC path in another partition, the partition for this edge can be changed to provide a DC path. After the change in the partition, the labels for the corresponding nodes and edges are updated and the DC path may be propagated to some other edges without a DC path. In the final splitting result, a DC path is made available for all the nodes in both sub-graphs.

[0034] FIGS. 4A through 4G illustrate an example of the above splitting algorithm for a simple circuit shown by a mesh graph in FIG. 4A. Node and edge labels are defined in the insert of FIG. 4B to include CONNECTED, UNCONNECTED, ZERO_UNCONNECTED, ONE_UNCONNECTED, ZERO_ONE_UNCONNECTED, ZERO_CONNECTED and ONE_CONNECTED, where "ZERO" and "ONE" represent two sub-graphs due to the splitting. The label's name describes the status of partition and DC path for the node. A node is "connected" when it is connected to a GND or VDD. Hence, the label "ZERO_UNCONNECTED" for a node means that the node has an edge in the sub-graph ZERO without a DC path. Other labels are self explanatory.

[0035] FIG. 4A shows the input undirected 6x6 mesh graph for a circuit where there are two VDD/GND nodes S(2,5) and S(5,2) and all nodes are assumed to be super nodes. FIGS. 4B-4F show the stepwise changes of splitting status for all the nodes and edges in the BFS partition stage. FIG. 4B shows the first step of the BFS process which starts from the VDD/GND nodes S(2,5) and S(5,2). The edges of nodes S(2,5) and S(5,2) are divided into two partitions evenly according to the balance rule. The labels for the edges and its neighbors also are updated at the end of the first step of the BFS partition.

[0036] Next, the edges for the nodes S(1,5), S(2,6), S(3,5), S(2,4), S(4,2), S(5,3), S(6,2) and S(5,1) are divided. The edges of nodes S(1,6) and S(6,1) belong to the same partition according to the degree rule. The edge between nodes S(3,5) and S(3,6) are assigned partition to avoid loop according to the loop rule. The result of these operations is shown in FIG. 4C.

[0037] FIG. 4D shows additional edges are divided into two partitions based on the rules. The edges far away from the input VDD/GND nodes S(2,5) and S(5,2) are more likely to have no DC path. Six nodes remain unconnected at this time. FIG. 4E shows that the edges of six unconnected nodes S(1,3), S(2,2), S(3,1), S(4,6), S(5,5) and S(6,4) in FIG. 4D are processed and only two nodes S(1,1) and S(6,6) are left unconnected. The BFS partition continues to the remaining two nodes S(1,1) and S(6,6). FIG. 4F shows the final partition result of the BFS partition stage. There are 32 edges and 28 edges in the two partitions, respectively, where 23 edges have

DC paths in their partition while 37 edges do not have DC paths.

[0038] Next, the post processing is performed to modify the partition to ensure the DC paths for all edges. FIG. 4G shows the final partition result after the post processing. Both sub-graphs in the final result have a tree/forest structure and there are 31 and 29 edges in the two partitions, respectively. The tree/forest structure greatly benefits the LU decomposition operations in the subsequent circuit simulation.

[0039] The above splitting algorithm can be shown be unconditionally stable. For the analysis of the error propagation, the inputs in the operator splitting formulation (5) can be ignfored. The two half steps can be combiend and reduced to a recursive formula:

$$X_{(k+1)} = \Lambda X_{(k)} \qquad (7)$$

where $\Lambda = (P_2 + S)^{-1} (P_1 - S)(P_1 + S)^{-1}(P_2 - S)$. In proof of the convergence, the norm as follows is used:

$$\|x\|_{s^{-1}} = \left(\chi^T S^{-1} \chi\right)^{1/2}.$$

The matrix S^{-1} is positive definite because the matrix S is positive definite and the inverse of a positive definite matrix remains to be positive definite. The following sections first state the theorem of the unconditional stability and then provide the proof of the statement by lemmas which follow the theorem.

[0040] ***Theorem 3.1:*** The operator splitting formula (5) is stable independent of the step size *h*.

[0041] ***Proof:*** Let $p(\Lambda) = \max(|\lambda_1(\Lambda)|)$, where $\lambda_i(\Lambda)$ is the i^{th} eigenvalue of matrix Λ. The proposed operator splitting approach is stable if $p(\Lambda) <= 1$.

[0042] From Lemma 3.4, we have the following

$$\left\|(P_1-S)(P_1+S)^{-1}\chi\right\|_{s^{-1}}\le\|\chi\|_{s^{-1}} \text{ and}$$
$$\left\|(P_2-S)(P_2+S)^{-1}\chi\right\|_{s^{-1}}\le\|\chi\|_{s^{-1}}.$$

Let $\rho\left(\tilde{\Lambda}\right)=(P_1-S)(P_1+S)^{-1}(P_2-S)(P_2+S)^{-1}$

From Lemma 3.2 and 3.3, we can deduce: $\rho(\Lambda)=\rho\left(\tilde{\Lambda}\right)\le 1$

Lemma 3.2:

$$\rho\left((P_2+S)^{-1}(P_1-S)(P_1+S)^{-1}(P_2-S)\right)=\rho\left((P_1-S)(P_1+S)^{-1}(P_2-S)(P_2+S)^{-1}\right)$$

Proof: It can be derived that $\rho(AB)=\rho(BA)$ if matrix A or B is nonsingular. Thus, the lemma can be proven by setting A = $(P_2 + S)^{-1}$ and B = $(P_1 - S)(P_1 + S)^{-1}$ $(P_2 - S)$.

Lemma 3.3: Given a real matrix *M*, if $\|M\chi\|_{s^{-1}}\le\gamma\|\chi\|_{s^{-1}}$ for all real χ, then $\rho(M)\le\gamma$.

The proof can be found in E. L. Wachspress and G. J. Habetler, "An alternating-direction¬implicit iteration technique," J. Soc. Ind. and Appl. Math. 8, 403-424(1960).

Lemma 3.4: $\left\|(P_i-S)(P_i+S)^{-1}\chi\right\|_{s^{-1}}^2\le\|\chi\|_{s^{-1}}^2$ for $i\in\{1,2\}$ and every real vector χ.

Proof: $\left\|(P_i-S)(P_i+S)^{-1}\chi\right\|_{s^{-1}}^2\le\|\chi\|_{s^{-1}}^2$ is equivalent to

$\left\|(P_i-S)(P_i+S)^{-1}\chi\right\|_{s^{-1}}^2\le\|\chi\|_{s^{-1}}^2$ where $y=(P_i+s)^{-1}\chi$

We expand the inequality expression according to the definition of the norm.

$$y^T\left(P_i^T-S^T\right)S^{-1}(P_i-S)y\le y^T\left(P_i^T+S^T\right)S^{-1}(P_i+S)y \qquad (8)$$

We expand the product terms and cancel the common items on the two sides of the inequality. The expression is reduced to:

$$y\left(P_i + P_i^T\right)y^T \geq 0 \quad (9)$$

which is true since $P_i + P_i^T$ is positive semidefinite for $i \in \{1,2\}$.

[0043] Though the general operator splitting approach is A-stable, the local truncation error still needs to be controlled below the error tolerance in order to ensure the accuracy. By estimating the local truncation error at each time point, we can dynamically adjust the time step to control the local truncation error.

[0044] Consider the system equation before the numerical integration,

$$\begin{bmatrix} C & 0 \\ 0 & L \end{bmatrix}\begin{bmatrix} \dot{V}(t) \\ \dot{I}(t) \end{bmatrix} = \begin{bmatrix} -G & A^T \\ -A & -R \end{bmatrix}\begin{bmatrix} V(t) \\ I(t) \end{bmatrix} + U(t) \quad (10)$$

Let $M = \begin{bmatrix} C & 0 \\ 0 & L \end{bmatrix}, N = \begin{bmatrix} -G & A^T \\ -A & -R \end{bmatrix}$ and ignore the input vector *U*. Equation (10) can be simplified as:

$$\begin{aligned} M\dot{X} &= NX \\ \dot{X} &= M^{-1}NX \end{aligned} \quad (11)$$

The exact analytic solution *x* with time step *h* can be derived as below:

$$\begin{aligned} X_{n+1} &= e^{M-1Nh}X_n \\ &= \left(1 + M^{-1}Nh + \frac{h^2(M^{-1}N)^2}{2} + \frac{h^3\left(M^{-1}N\right)^3}{6} + O\left(h^4\right)\right)X_n \end{aligned} \quad (12)$$

The general operator splitting approach can also be formulated as:

$$\frac{M}{h}\left(\hat{X}_{n+1} - X_n\right) = N_1\hat{X}_{n+1} + N_2X_n \quad (13)$$

where $N = N_1 + N_2$, N_1 represents the partition applied Backward Euler and N_1 denotes the partition applied forward Euler integration method.

[0045] The analytic solution of operator splitting approach is derived as below:

$$\left(\frac{M}{h}-N_1\right)\hat{X}_{n+1}=\left(\frac{M}{h}+N_2\right)X_n \tag{14}$$

$$\hat{X}_{n+1}=\left[I+hM^{-1}N+h^2M^{-1}N_1M^{-1}N+O\left(h^3\right)\right]X_n \tag{15}$$

The local truncation error (LTE) is the difference of operator splitting solution and exact solution x:

$$LTE=\left\|h^2M^{-1}\left(\frac{N}{2}-N_1\right)\dot{X}_n+O\left(h^3\right)\right\| \tag{16}$$

The local truncation error at each time step should not exceed the error tolerance. If the high order terms of local truncation error are ignored, the time step when forward Euler integration is applied to partition corresponding to N_1 can be estimated as:

$$h_1<\sqrt{\frac{ErrorTolerance}{\left\|M^{-1}\left(\frac{N}{2}-N_1\right)\dot{X}_n\right\|}} \tag{17}$$

Similarly, the time step when forward Euler integration is applied to partition corresponding to N_2 is estimated as:

$$h_2<\sqrt{\frac{ErrorTolerance}{\left\|M^{-1}\left(\frac{N}{2}-N_2\right)\dot{X}_n\right\|}} \tag{18}$$

The new time step h is twice of the minimum time step of each partition:

$$h = 2\min\left(h_1, h_2\right) \qquad (19)$$

[0046] The above splitting algorithm was implemented in C programming language and tested against Berkeley SPICE3 using BSIM3 models for transistor devices. Convergence and accuracy are guaranteed. Examples were tested on a Linux machine with 2.6 GHz CPU and 4 Gigabytes memory.

[0047] A number of RLC power networks with nonlinear current sinks ranging from 11k nodes to 160k nodes were tested. Various transistor gates draw current from the power networks. Those power networks are approximately in mesh structures. The splitting algorithm results in very limited nonzero fill-ins and the linear runtime of the splitting algorithm was. The CPU runtime and speedup are given in Table I. One or two orders of magnitude speedup (8.1x to 58.2x) against SPICE3 is obtained. The transient waveform circuit3 is given in FIG. 5A. The tests only replaced the LU decomposition procedure inside the SPICE3, other overhead such as device evaluation and dynamic time step control took more than 30% of the total runtime and therefore limited the overall speedup during our tests.

The power and clock network tests were done for an RLC power ground network and a two-level H-tree clock. FIG. 5B shows the voltage drop at one node of the power network. Transient simulation of 10 ns is completed in 649.5 seconds, which is 18.5 times faster than SPICE3 as shown in Table I.

A large RC power network with 0.6 million nodes and an irregular structure was also tested where some nodes have thousands of neighbors. The switching activities that draw current from the power network were modeled as piecewise linear current waveform. Berkeley SPICE3 failed to execute because of the memory size and computation time problem. The operator splitting approach finished the transient analysis of

10 ns in just 4083 seconds. FIG. 5C illustrates the voltage drop of a node on the power network.

[0048] Two 1K and 10K cells ASIC designs were also tested to demonstrate the splitting algorithm's ability of handling transistor dominated nonlinear circuits. The 1k cell circuit had 10,200 nodes and 6,500 transistors. The 10k cell circuit had 123,600 nodes and 69,000 transistors. It was assumed that ideal power and ground supply were provided in those RC examples. The splitting algorithm took 415.9 seconds for the 1K cell circuit and 3954.7 seconds for the 10K cell circuit to finish 20-ns transient simulations. The speedup over SPICE3 is 5.1x and 11.2x for these two examples (Table I). Accurate waveform match was observed for those two examples. FIG. 5D shows the transient waveform of one gate output in the 1K cell design.

TABLE I

TRANSIENT SIMULATION RUNTIME

Examples	circuit1	circuit2	circuit3	circuit4	Power and Clock Network	Large Power Network	1K-cell	10K-cell
#Nodes	11,203	41,321	92,360	160,657	29,100	615,446	10,200	123,600
#Transistors	74	512	1,108	2,130	720	0	6,500	69,000
Simulation Period	10ns	10ns	10ns	10ns	10ns	10ns	20ns	20ns
SPICE3 (sec)	602.44	8268.92	39612.32	N/A	12015	N/A	2121	44293
Operator Splitting (sec)	74.64	305.38	681.18	1356.21	649.5	4083.7	415.9	3954.7
Speedup	8.1x	27.1x	58.2x	N/A	18.5x	N/A	5.1x	11.2x

[0049] In implementations, the above described techniques and their variations may be implemented as computer software instructions. Such software instructions may be stored in an article with one or more machine-readable storage media that are not connected to a computer, or stored in one or more machine-readable storage devices connected to one or more computers either directly or via a communication link. In operation, the instructions are executed by, e.g., one or more computer processors, to cause the machine to perform the described functions and operations for circuit analysis.

[0050] Only a few implementations are disclosed. However, it is understood that variations and enhancements may be made.

Claims

What is claimed is:

1. A method for analyzing an integrated circuit, comprising:

representing the integrated circuit with a graph;

from one or more nodes connected to GND or VDD in the graph, conducting a search for nodes in the graph;

dividing edges of nodes from the search to form two subgraphs that respectively represent two sub circuits of the integrated circuit;

modifying the two subgraphs to ensure each node of each subgraph to have a DC path to GND or VDD; and

analyzing the two subgraphs.

2. The method as in claim 1, wherein a capacitor is not divided between the two subgraphs and is duplicated in the two subgraphs.

3. The method as in claim 1, wherein an inductor is not divided between the two subgraphs and is duplicated in the two subgraphs.

4. The method as in claim 1, wherein a transistor is not divided between the two subgraphs and is duplicated in the two subgraphs.

5. The method as in claim 1, wherein a gate having a plurality of transistors is not divided between the two subgraphs and is duplicated in the two subgraphs.

6. The method as in claim 1, wherein edges in one branch are assigned to the same subgraph.

7. The method as in claim 1, wherein edges of a node whose degree is two are assigned to the same subgraph.

8. The method as in claim 1, wherein each subgraph is free of a loop.

9. The method as in claim 1, wherein a number of loops for nodes in each subgraph is minimized.

10. The method as in claim 1, wherein resistors of the integrated circuit are evenly divided between the two subgraphs.

11. The method as in claim 1, further comprising applying a direct matrix conversion method to analyze each of the two subgraphs.

12. The method as in claim 1, wherein the search is a breadth first search.

13. A method for analyzing an integrated circuit, comprising:

splitting the integrated circuit into two sub circuits which are in or close to tree or forest strcutrues and have a DC path to GND or VDD for each node; and

applying a direct matrix inversion method to a circuit matrix equation for each of the two sub circuits to find solutions to the sub circuits.

14. The method as in claim 13, wherein the splitting is configured to reduce a total number of non-zero fill-ins for the matrices for the two sub circuits to be less than a total number of non-zero fill-ins for the matrix for the integrated circuit prior to the splitting.

15. A method for analyzing an integrated circuit, comprising:

representing the integrated circuit with a graph;

dividing the graph into two subgraphs that respectively represent two sub circuits of the integrated circuit, where a resistor is divided evenly between the two subgraphs while a transistor, a transistor device, a capacitor, or an inductor is duplicated in the two subgraphs, and wherein each node of each subgraph has a DC path to GND or VDD; and

finding solutions to the two subgraphs for analyzing the integrated circuit.

16. The method as in claim 15, wherein edges in one branch are assigned to the same subgraph.

17. The method as in claim 15, wherein edges of a node whose degree is two are assigned to the same subgraph.

18. The method as in claim 15, wherein each subgraph is free of a loop.

19. The method as in claim 15, wherein a number of loops for nodes in each subgraph is minimized.

20. An article comprising at least one machine-readable medium that stores machine-executable instructions, the instructions causing a machine to:

represent the integrated circuit with a graph;

from one or more nodes connected to GND or VDD in the graph, conduct a search for nodes in the graph;

divide edges of nodes from the search to form two subgraphs that respectively represent two sub circuits of the integrated circuit;

modify the two subgraphs to ensure each node of each subgraph to have a DC path to GND or VDD; and

analyze the two subgraphs.

21. The article as in claim 20, wherein a capacitor is not divided between the two subgraphs and is duplicated in the two subgraphs.

22. The article as in claim 20, wherein an inductor is not divided between the two subgraphs and is duplicated in the two subgraphs.

23. The article as in claim 20, wherein a transistor is not divided between the two subgraphs and is duplicated in the two subgraphs.

24. The article as in claim 20, wherein a gate having a plurality of transistors is not divided between the two subgraphs and is duplicated in the two subgraphs.

25. The article as in claim 20, wherein edges in one branch are assigned to the same subgraph.

26. The article as in claim 20, wherein edges of a node whose degree is two are assigned to the same subgraph.

27. The article as in claim 20, wherein each subgraph is free of a loop.

28. The article as in claim 20, wherein a number of loops for nodes in each subgraph is minimized.

29. The article as in claim 20, wherein resistors of the integrated circuit are evenly divided between the two subgraphs.

30. The article as in claim 20, further comprising applying a direct matrix conversion method to analyze each of the two subgraphs.

31. The article as in claim 20, wherein the search is a breadth first search.

FIG. 1A

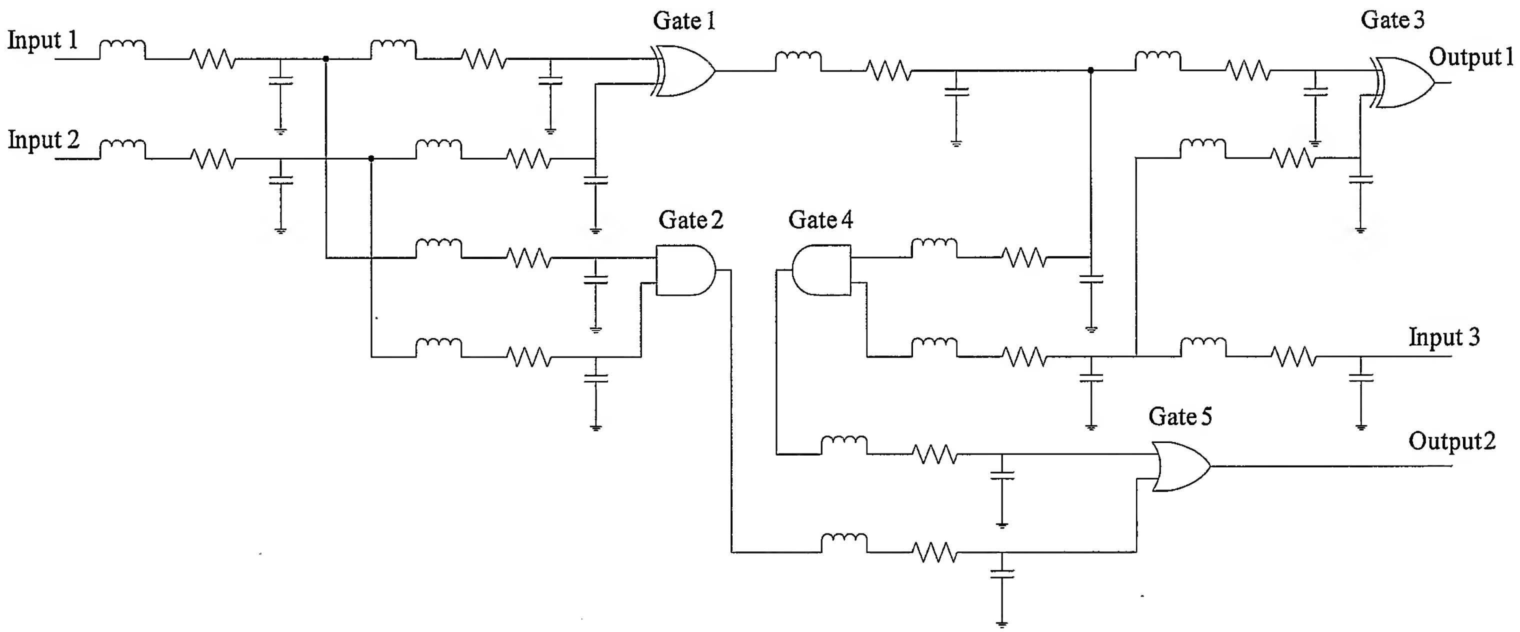

Input 1
Input 2
Gate 1
Gate 2
Gate 4
Gate 3
Output 1
Input 3
Gate 5
Output2

FIG. 1B

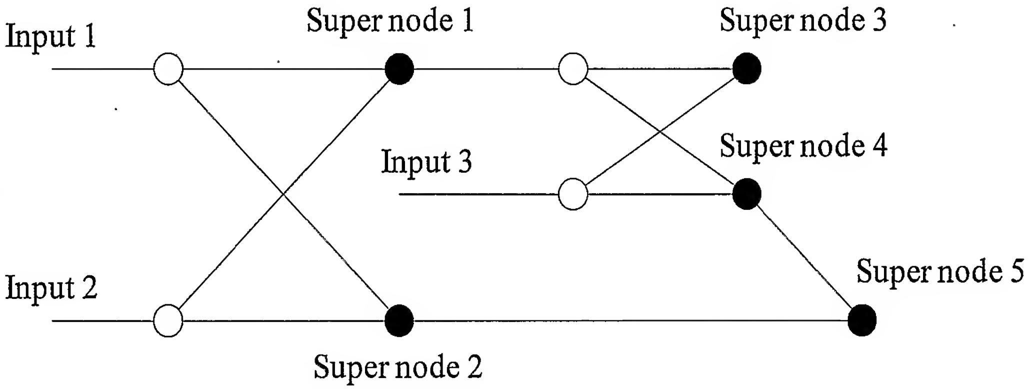

Input 1
Input 2
Super node 1
Super node 2
Input 3
Super node 3
Super node 4
Super node 5

FIG. 2A

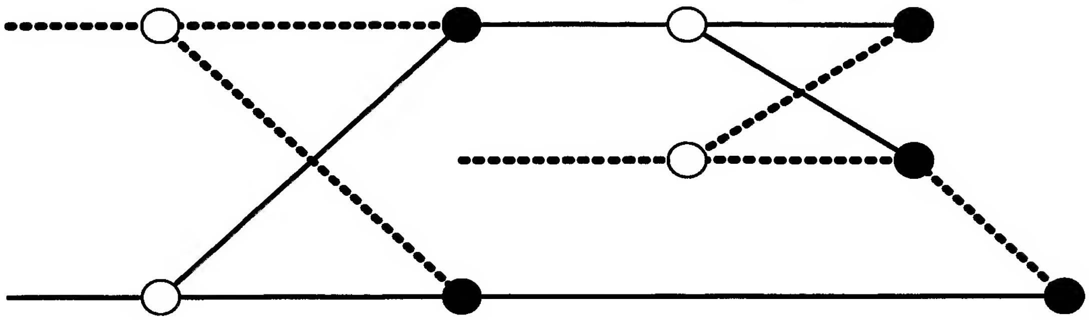


FIG. 2B

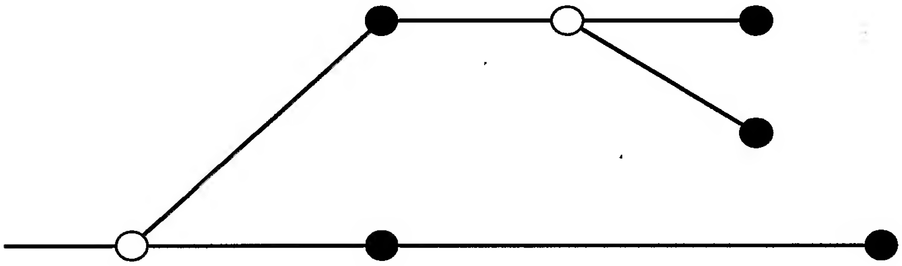


FIG. 2C

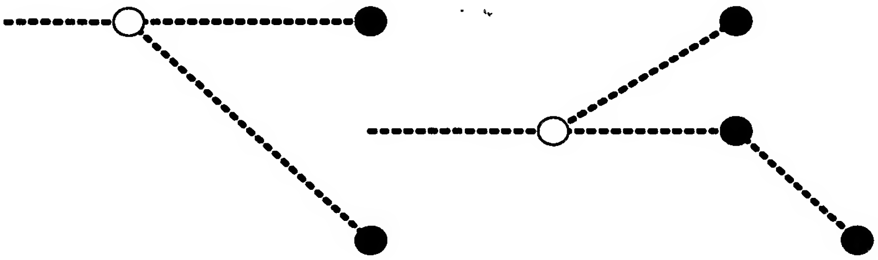


FIG. 3

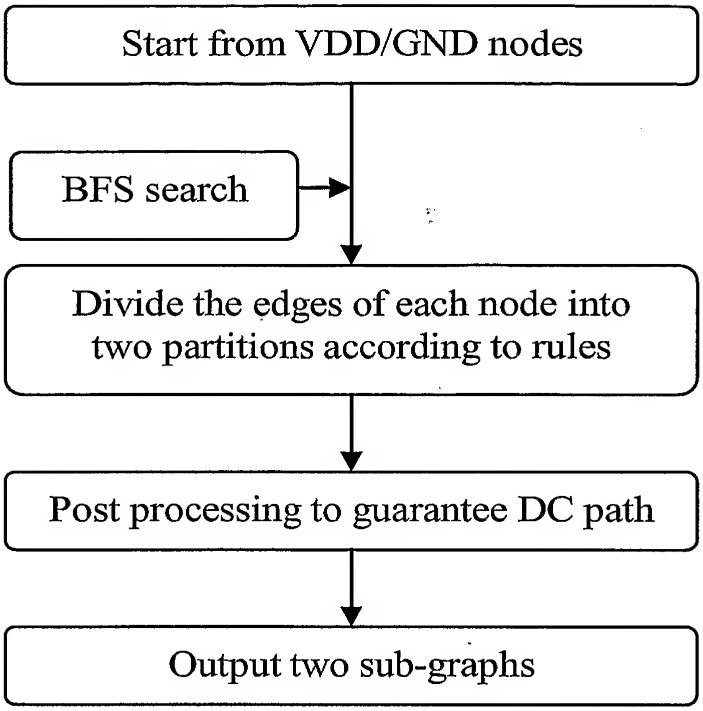

Start from VDD/GND nodes
BFS search
Divide the edges of each node into two partitions according to rules
Post processing to guarantee DC path
Output two sub-graphs

FIG. 4A

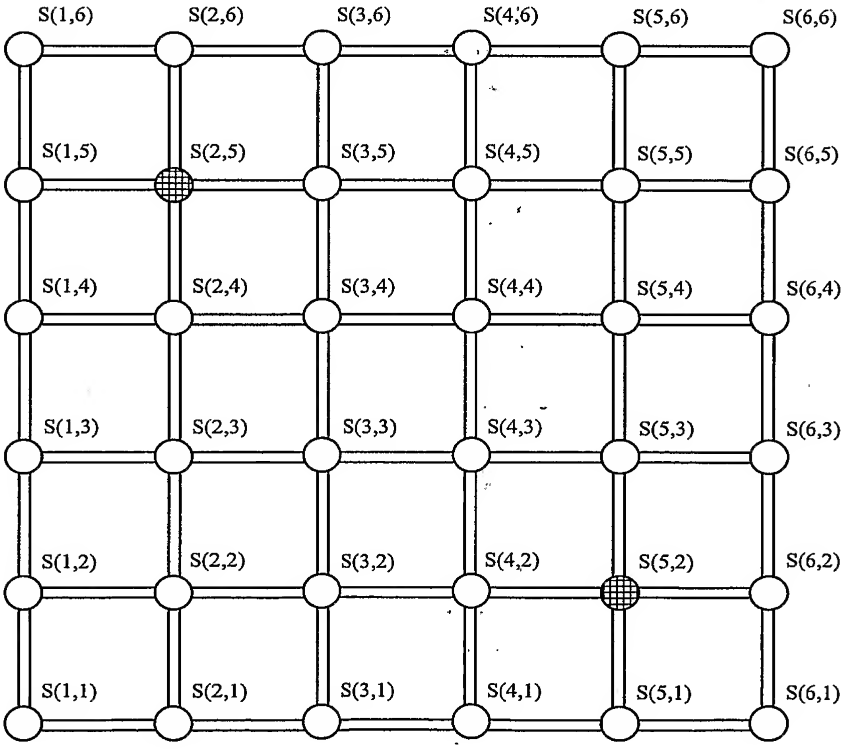

S(1,6)
S(2,6)
S(3,6)
S(4,6)
S(5,6)
S(6,6)
S(1,5)
S(2,5)
S(3,5)
S(4,5)
S(5,5)
S(6,5)
S(1,4)
S(2,4)
S(3,4)
S(4,4)
S(5,4)
S(6,4)
S(1,3)
S(2,3)
S(3,3)
S(4,3)
S(5,3)
S(6,3)
S(1,2)
S(2,2)
S(3,2)
S(4,2)
S(5,2)
S(6,2)
S(1,1)
S(2,1)
S(3,1)
S(4,1)
S(5,1)
S(6,1)

FIG. 4B

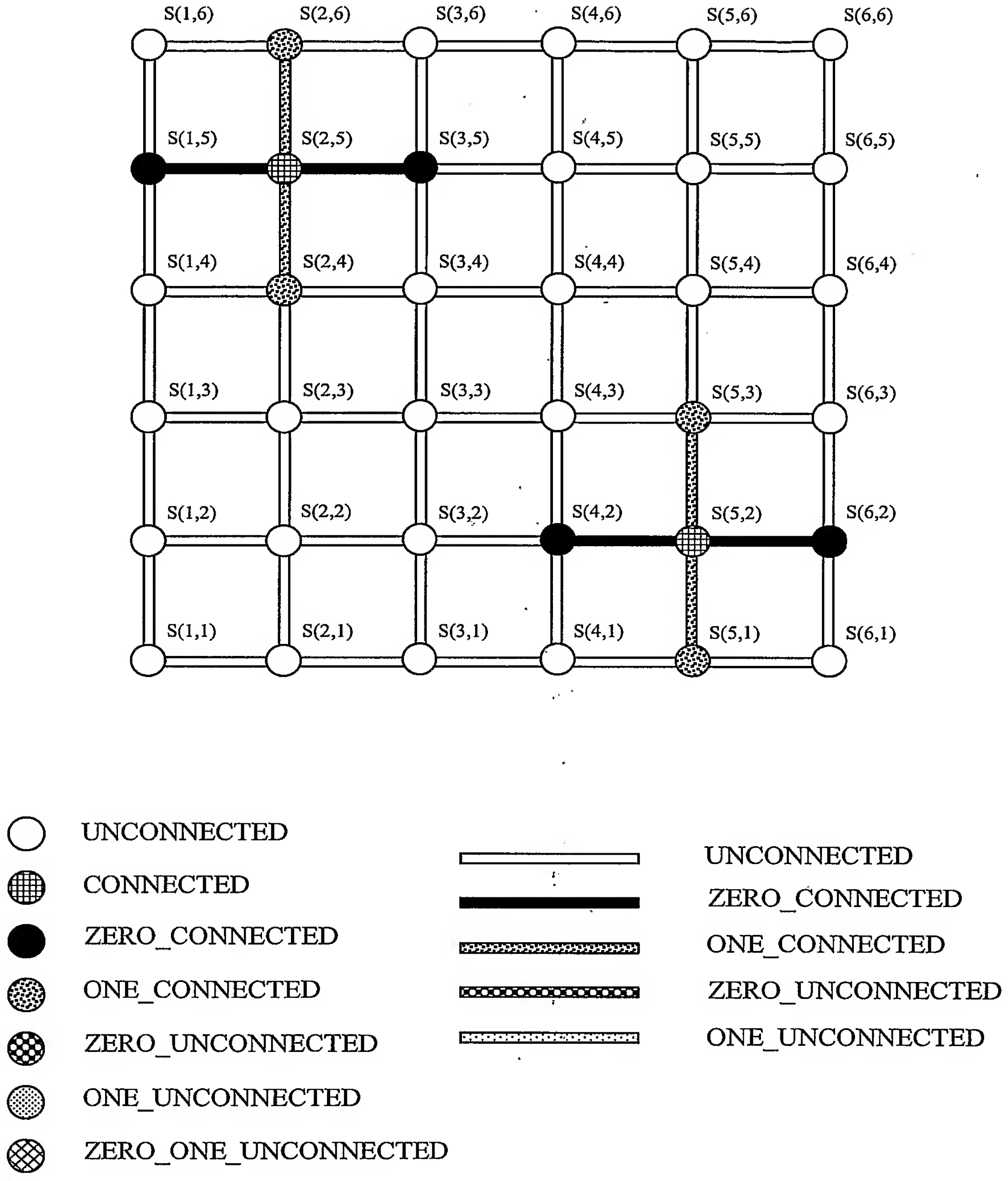

S(1,6)
S(2,6)
S(3,6)
S(4,6)
S(5,6)
S(6,6)
S(1,5)
S(2,5)
S(3,5)
S(4,5)
S(5,5)
S(6,5)
S(1,4)
S(2,4)
S(3,4)
S(4,4)
S(5,4)
S(6,4)
S(1,3)
S(2,3)
S(3,3)
S(4,3)
S(5,3)
S(6,3)
S(1,2)
S(2,2)
S(3,2)
S(4,2)
S(5,2)
S(6,2)
S(1,1)
S(2,1)
S(3,1)
S(4,1)
S(5,1)
S(6,1)
UNCONNECTED
CONNECTED
ZERO_CONNECTED
ONE_CONNECTED
ZERO_UNCONNECTED
ONE_UNCONNECTED
ZERO_ONE_UNCONNECTED
UNCONNECTED
ZERO_CONNECTED
ONE_CONNECTED
ZERO_UNCONNECTED
ONE_UNCONNECTED

FIG. 4C

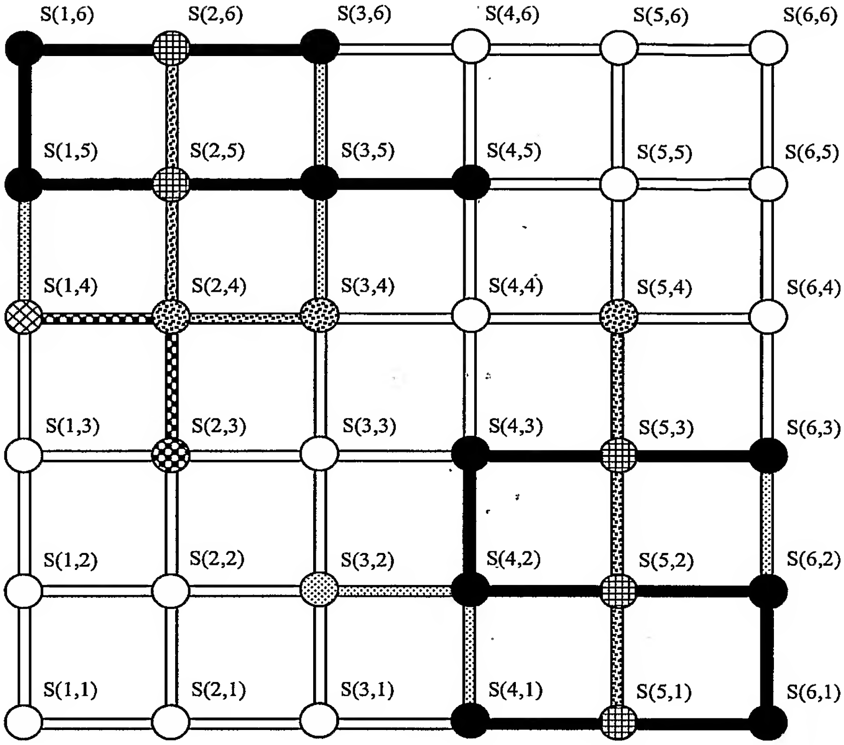

S(1,6)
S(2,6)
S(3,6)
S(4,6)
S(5,6)
S(6,6)
S(1,5)
S(2,5)
S(3,5)
S(4,5)
S(5,5)
S(6,5)
S(1,4)
S(2,4)
S(3,4)
S(4,4)
S(5,4)
S(6,4)
S(1,3)
S(2,3)
S(3,3)
S(4,3)
S(5,3)
S(6,3)
S(1,2)
S(2,2)
S(3,2)
S(4,2)
S(5,2)
S(6,2)
S(1,1)
S(2,1)
S(3,1)
S(4,1)
S(5,1)
S(6,1)

FIG. 4D

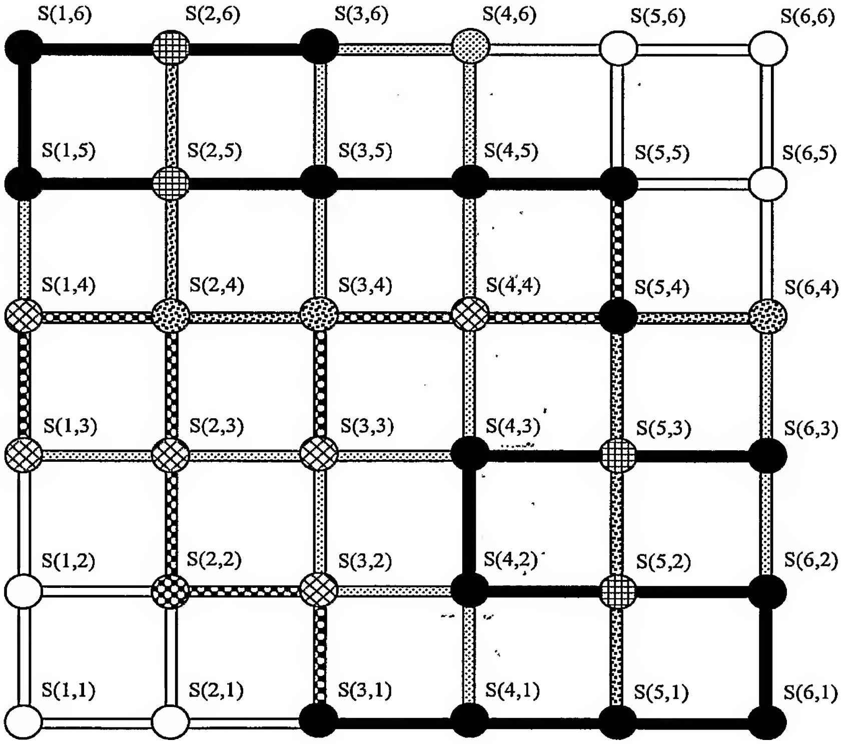

S(1,6)
S(2,6)
S(3,6)
S(4,6)
S(5,6)
S(6,6)
S(1,5)
S(2,5)
S(3,5)
S(4,5)
S(5,5)
S(6,5)
S(1,4)
S(2,4)
S(3,4)
S(4,4)
S(5,4)
S(6,4)
S(1,3)
S(2,3)
S(3,3)
S(4,3)
S(5,3)
S(6,3)
S(1,2)
S(2,2)
S(3,2)
S(4,2)
S(5,2)
S(6,2)
S(1,1)
S(2,1)
S(3,1)
S(4,1)
S(5,1)
S(6,1)

FIG. 4E

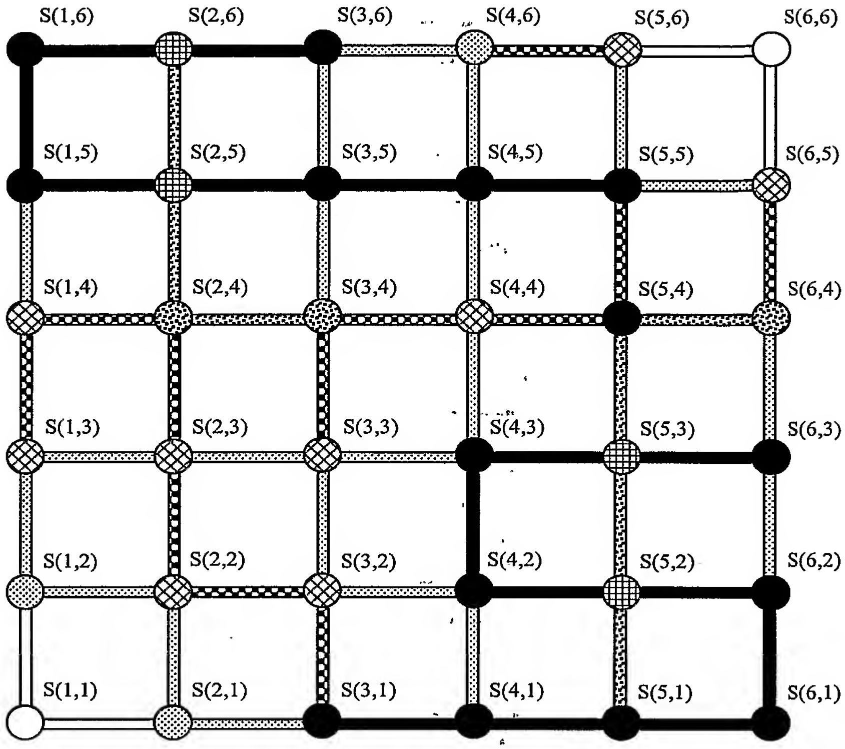

S(1,6)
S(2,6)
S(3,6)
S(4,6)
S(5,6)
S(6,6)
S(1,5)
S(2,5)
S(3,5)
S(4,5)
S(5,5)
S(6,5)
S(1,4)
S(2,4)
S(3,4)
S(4,4)
S(5,4)
S(6,4)
S(1,3)
S(2,3)
S(3,3)
S(4,3)
S(5,3)
S(6,3)
S(1,2)
S(2,2)
S(3,2)
S(4,2)
S(5,2)
S(6,2)
S(1,1)
S(2,1)
S(3,1)
S(4,1)
S(5,1)
S(6,1)

FIG. 4F

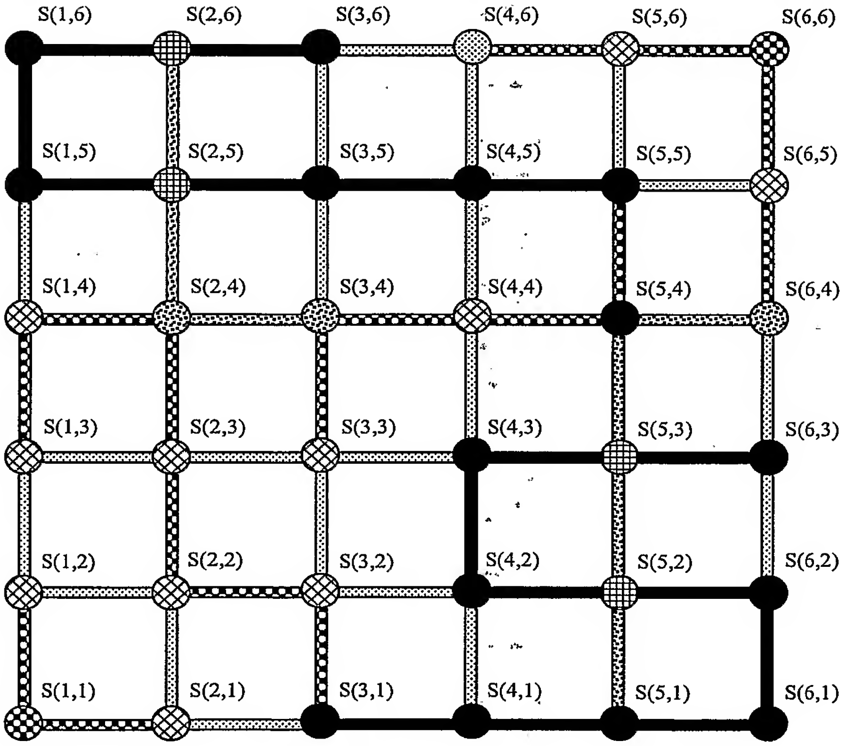

S(1,6)
S(2,6)
S(3,6)
S(4,6)
S(5,6)
S(6,6)
S(1,5)
S(2,5)
S(3,5)
S(4,5)
S(5,5)
S(6,5)
S(1,4)
S(2,4)
S(3,4)
S(4,4)
S(5,4)
S(6,4)
S(1,3)
S(2,3)
S(3,3)
S(4,3)
S(5,3)
S(6,3)
S(1,2)
S(2,2)
S(3,2)
S(4,2)
S(5,2)
S(6,2)
S(1,1)
S(2,1)
S(3,1)
S(4,1)
S(5,1)
S(6,1)

FIG. 4G

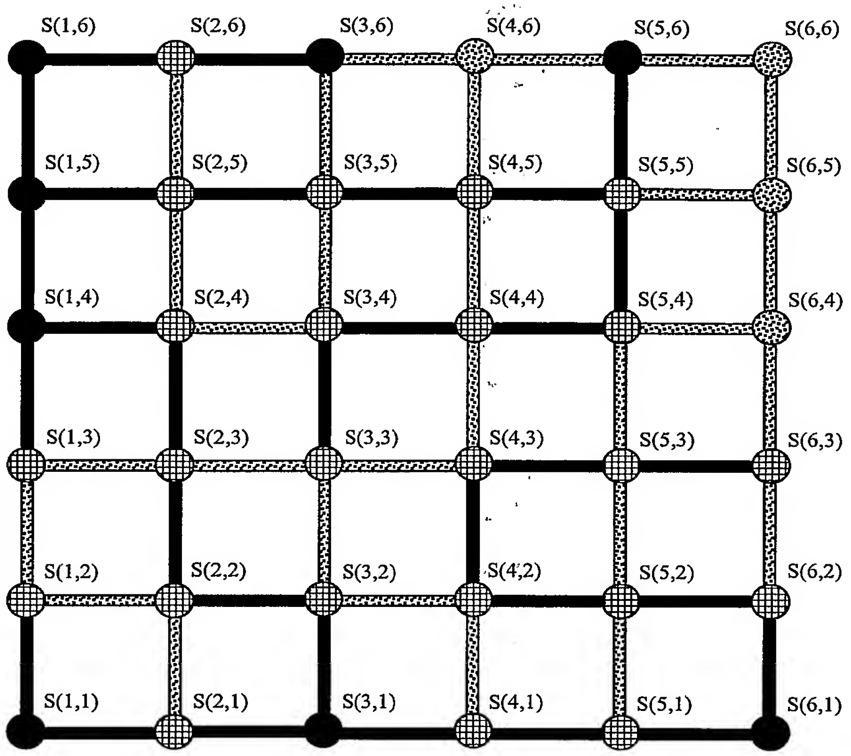

S(1,6)
S(2,6)
S(3,6)
S(4,6)
S(5,6)
S(6,6)
S(1,5)
S(2,5)
S(3,5)
S(4,5)
S(5,5)
S(6,5)
S(1,4)
S(2,4)
S(3,4)
S(4,4)
S(5,4)
S(6,4)
S(1,3)
S(2,3)
S(3,3)
S(4,3)
S(5,3)
S(6,3)
S(1,2)
S(2,2)
S(3,2)
S(4,2)
S(5,2)
S(6,2)
S(1,1)
S(2,1)
S(3,1)
S(4,1)
S(5,1)
S(6,1)

FIG. 5A

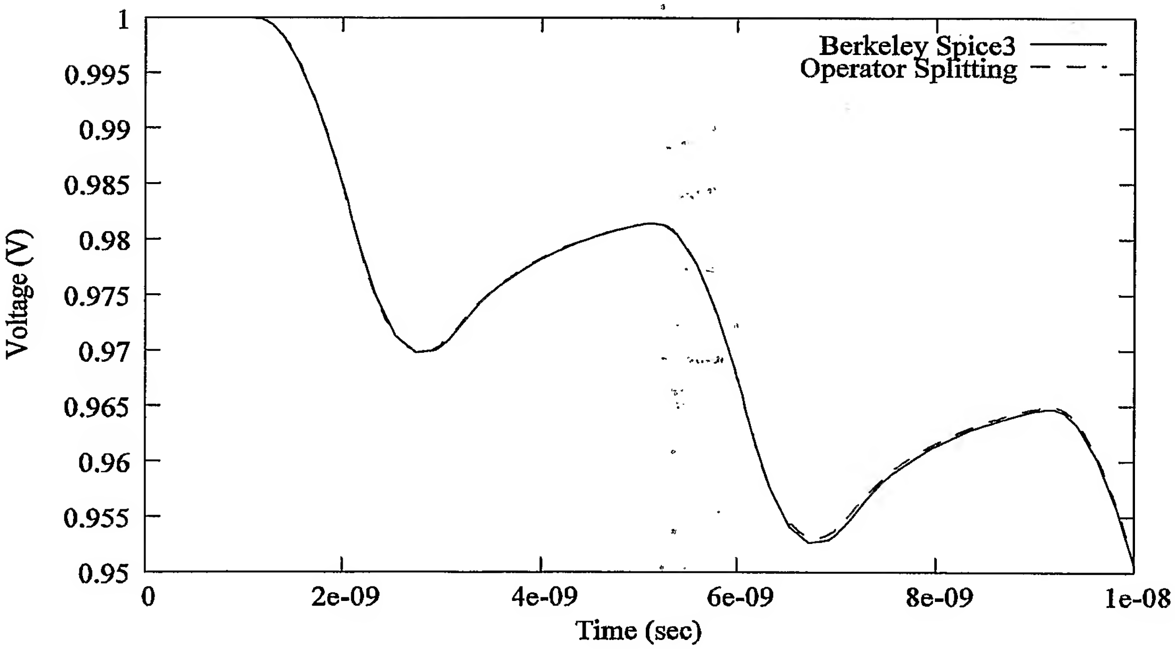

Berkeley Spice3
Operator Splitting
Voltage (V)
1
0.995
0.99
0.985
0.98
0.975
0.97
0.965
0.96
0.955
0.95
0
2e-09
4e-09
6e-09
8e-09
1e-08
Time (sec)

FIG. 5B

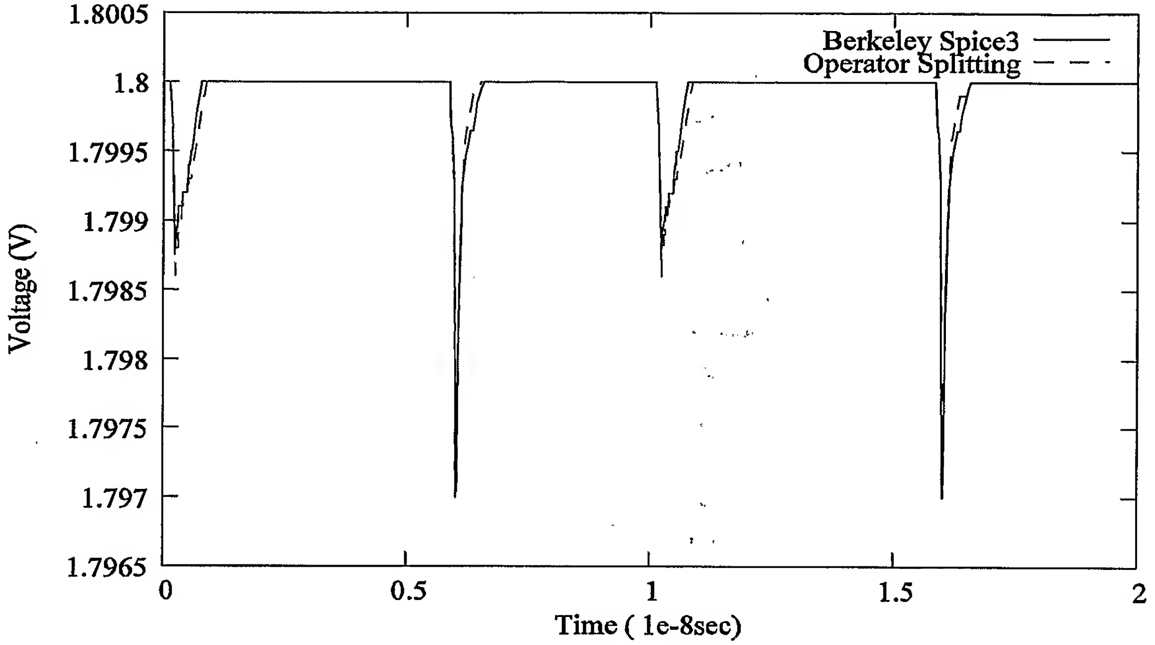

1.8005
1.8
1.7995
1.799
1.7985
1.798
1.7975
1.797
1.7965
Voltage (V)
0
0.5
1
1.5
2
Time (1e-8sec)
Berkeley Spice3
Operator Splitting

FIG. 5C

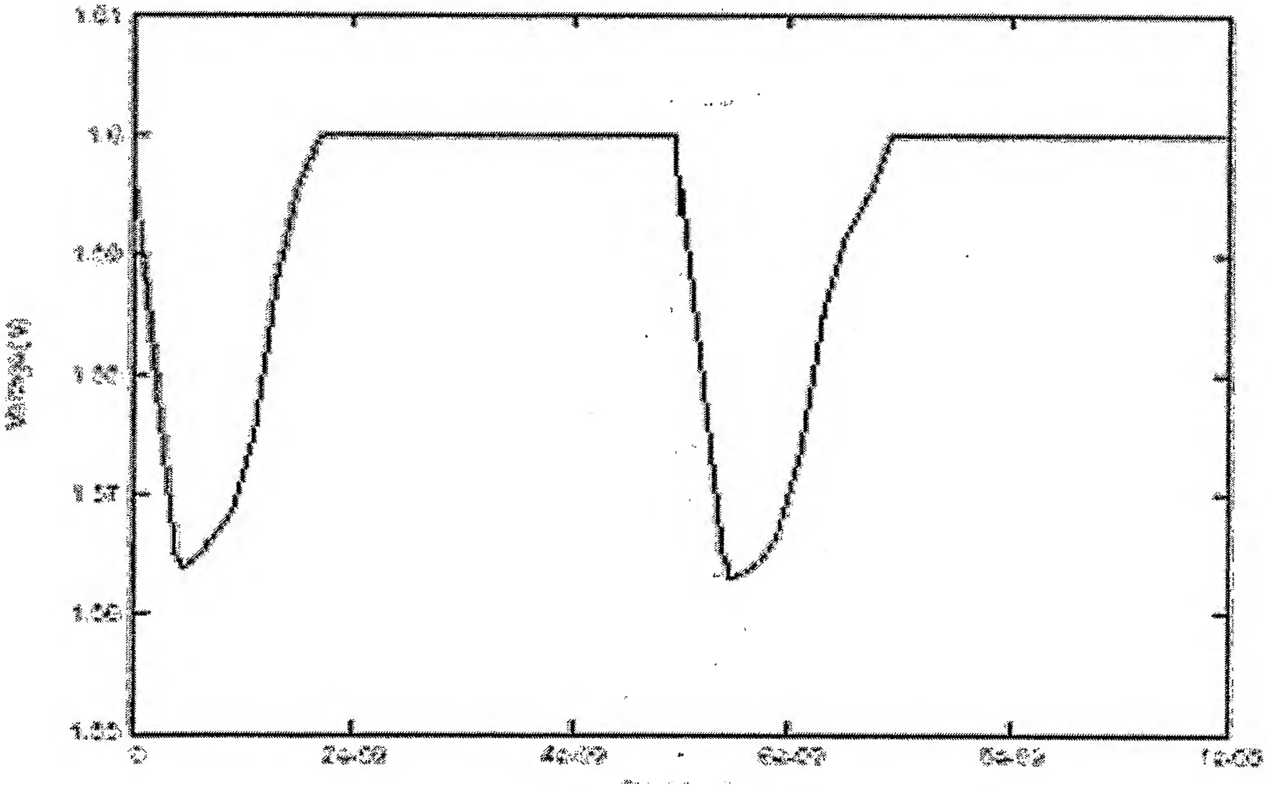
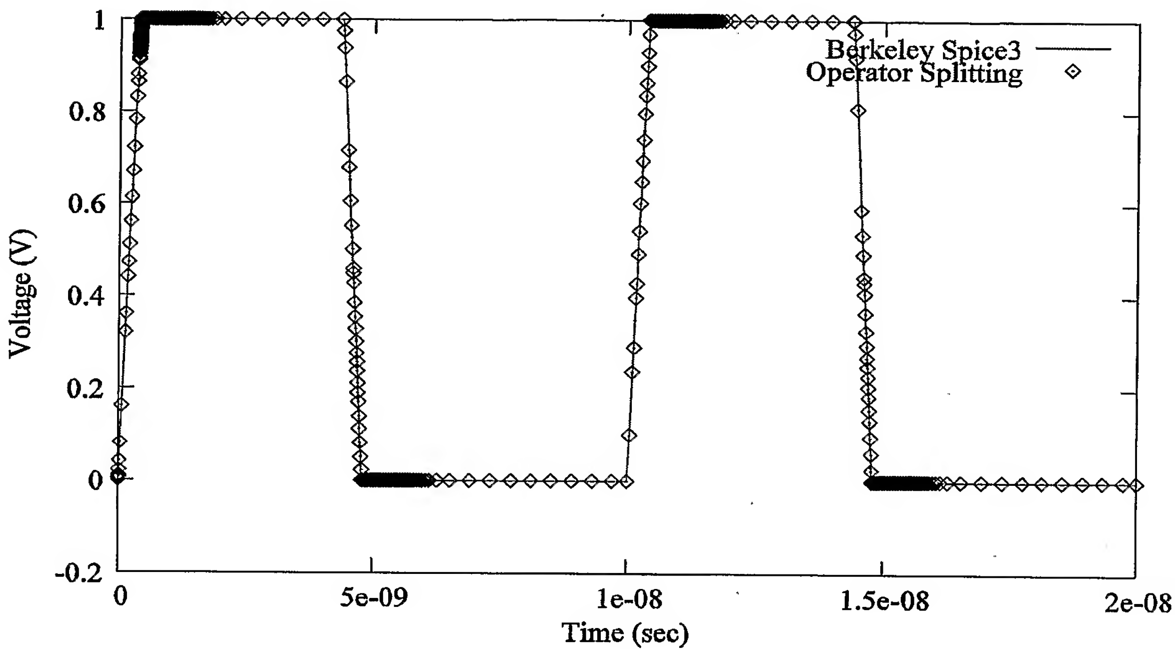

1.61
1.6
1.59
1.58
1.57
1.56
1.55
Voltage (V)
0
2e-09
4e-09
6e-09
8e-09
1e-08

FIG. 5D


Berkeley Spice3
Operator Splitting ◇
1
0.8
0.6
0.4
0.2
0
-0.2
Voltage (V)
0
5e-09
1e-08
1.5e-08
2e-08
Time (sec)

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US05/20242

A. CLASSIFICATION OF SUBJECT MATTER

IPC(7) : G06F 17/50
US CL : 702/081; 716/004

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
U.S. : 702/081; 716/004, 005, 006; 703/14-16

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
STIC for publications on circuit analysis and SPICE programs.

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EAST for key word and terms

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 6,557,148 B1 (Nishida et al.) 29 April 2003 (29.04.2003), see entire patent.	1-31.
A	US 6,308,300 B1 (Bushnell et al.) 23 October 2001 (23.10.2001), see entire patent.	1-31.

□ Further documents are listed in the continuation of Box C. □ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search 19 December 2005 (19.12.2005)	Date of mailing of the international search report 30 DEC 2005
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US Commissioner for Patents P.O. Box 1450 Alexandria, Virginia 22313-1450 Facsimile No. (571) 273-3201	Authorized officer Victor J. Taylor Sylvia Dell For Telephone No. 571-272-1750